

Технически Университет - София

Курсов Проект

Дисциплина:

База Данни

Тема:

Проектиране на релационна база от данни

Проверил/а:

Разработил:

Лъчезар Христов Спириев

Факултет: Телекомуникации

Специалност: Телекомуникации

Ф-н: 111221054

Група: 54



ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ

ФАКУЛТЕТ ПО ТЕЛЕКОМУНИКАЦИИ

Катедра „Комуникационни мрежи“
Период на задаване: **20.03.2023 ÷ 24.03.2023** Образователна степен: бакалавър
Период на предаване и защита: **02.05.2023 ÷ 12.05.2023** Специалност: Телекомуникации

ЗАДАНИЕ ЗА КУРСОВ ПРОЕКТ ПО БАЗИ ДАННИ

Вариант 14

ТЕМА: **Проектиране на релационна база от данни**

Студент: Лъчезар Христов Спириев **Курс: II** **Група: 54** **Фак. № 111221054**

1. Описание на инженерната задача:
Да се проектира релационна база от данни, описваща курсове за обучение, периодите от време, през които се провеждат курсовете за обучение и преподавателите, провеждащи курсовете за обучение. Да се разработят необходимите модели на базата данни като се определят обектите, атрибутите на съответните обекти и връзките между обектите. На базата на разработените модели да се създаде релационната база от данни и необходимите таблици. Да се гарантира цялостност на данните. Да се демонстрира функционалността на разработената база от данни като се съставят необходимите заявки за въвеждане и извличане на данни. Резултатите от изпълнението на съответните заявки да се визуализират в табличен вид и да се валидират.

2. Изходни данни: За СУБД да се използва MySQL. Данните да са структурирани в трета нормална форма.

3. Необходима информация и входни данни:
Всеки курс има уникален номер (course id), име (course name) и брой часове (course hours). Всеки период от време има уникален номер (time period id), начална дата (time period start), крайна дата (time period end) и часова ставка (time period hourly rate). Всеки преподавател има уникален номер (teacher id) и име (teacher name). Един курс може да се провежда в различни периоди от време, но в рамките на един период от време един курс се провежда еднократно. Един курс в рамките на един период от време може да се провежда от един или повече преподавателя. Един и същи курс в различните периоди от време може да се провежда от различни преподаватели. Един преподавател може да провежда различни курсове в различни периоди от време. Заплащането на преподавателите се определя в зависимост от часовата ставка на дадения период от време (time period hourly rate) и броя на проведените часове за дадения курс в рамките на дадения период от време (hours per teacher).

time period id	time period start	time period end	time period hourly rate	course id	course name	course hours	teacher id	teacher name	hours per teacher	total per teacher
1	06.02.2023	02.04.2023	120.00 лв.	1	Amazon AWS	42	1	Боряна Борисова	12	1 440.00 лв.
							2	Даниела Димитрова	30	3 600.00 лв.
				2	Microsoft Azure	36	1	Боряна Борисова	10	1 200.00 лв.
							3	Елена Емилова	26	3 120.00 лв.
				3	Google Cloud Platform	30	2	Даниела Димитрова	14	1 680.00 лв.
							3	Елена Емилова	16	1 920.00 лв.
Total										**12 960.00 лв.**
2	10.04.2023	04.06.2023	140.00 лв.	1	Amazon AWS	42	2	Даниела Димитрова	13	1 820.00 лв.
							3	Елена Емилова	29	4 060.00 лв.
				2	Microsoft Azure	36	1	Боряна Борисова	11	1 540.00 лв.
							2	Даниела Димитрова	25	3 500.00 лв.
				3	Google Cloud Platform	30	1	Боряна Борисова	16	2 240.00 лв.
							3	Елена Емилова	14	1 960.00 лв.
Total										**15 120.00 лв.**
Grand Total										**28 080.00 лв.**

Научен ръководител: /доц. д-р Павлина. Колева/

ВАЖНО: Курсовият проект се разработва спрямо изискванията и указанията!

Съдържание на курсовия проект.

Част.1: Проектиране, моделиране и описание

1.1 Създаване на ER (Entity-Relationship) модел
1.2 Изготвяне на проект на релационната база от данни
1.3 Разработване на EER (Enhanced Entity Relationship) модел
1.4 Описание на таблиците и връзките между таблиците
1.4.1 Описание на таблиците
1.4.2 Описание на връзките между таблиците

Част.2: SQL – Създаване на базата от данни и таблиците към нея. Въвеждане на данни

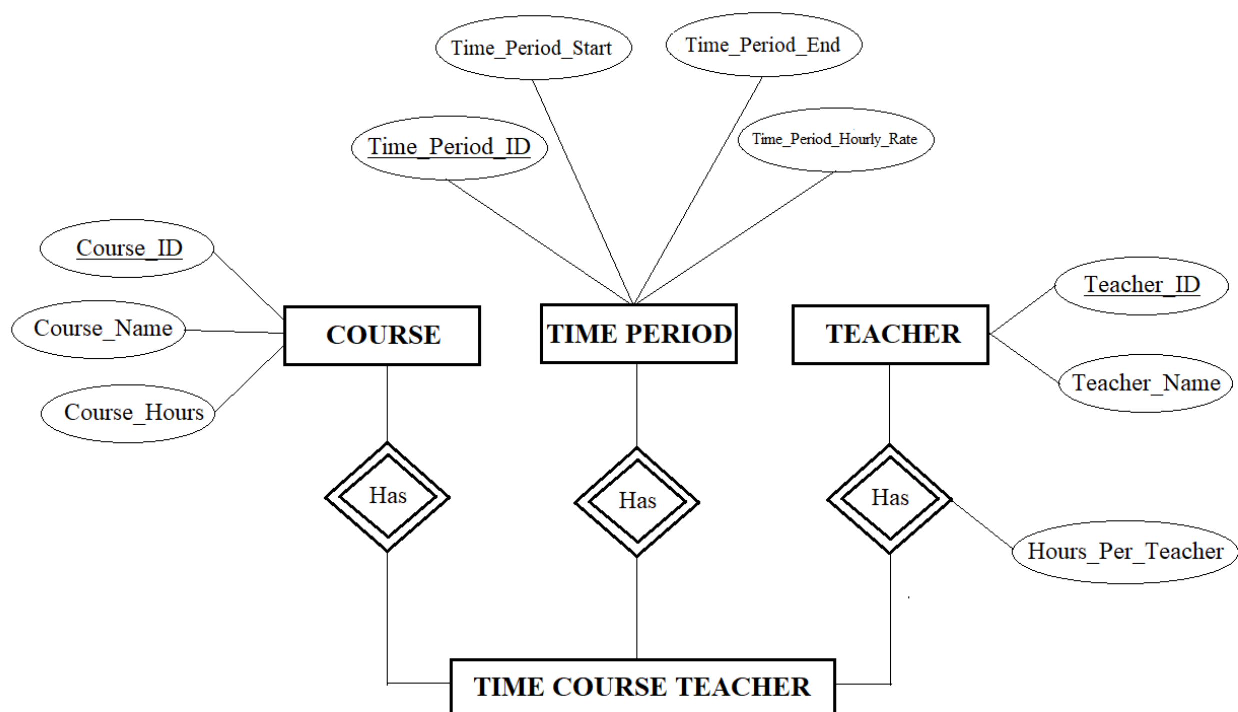
2.1 SQL заявка за създаване на базата от данни
2.2 SQL заявки за създаване на всяка таблица от базата данни
2.3 SQL заявки за въвеждане на данни във всяка една от таблиците

Част.3: SQL – Създаване на заявки по зададени условия и резултати от тяхното изпълнение

3.1 Пълни данни от всяка една таблица
3.2 Данни за курсовете за обучение
3.3 Данни за преподавателите, провеждащи курсовете за обучение
3.4 Данни за периодите от време, през които се провеждат курсовете за обучение
3.5 Съхранени процедури

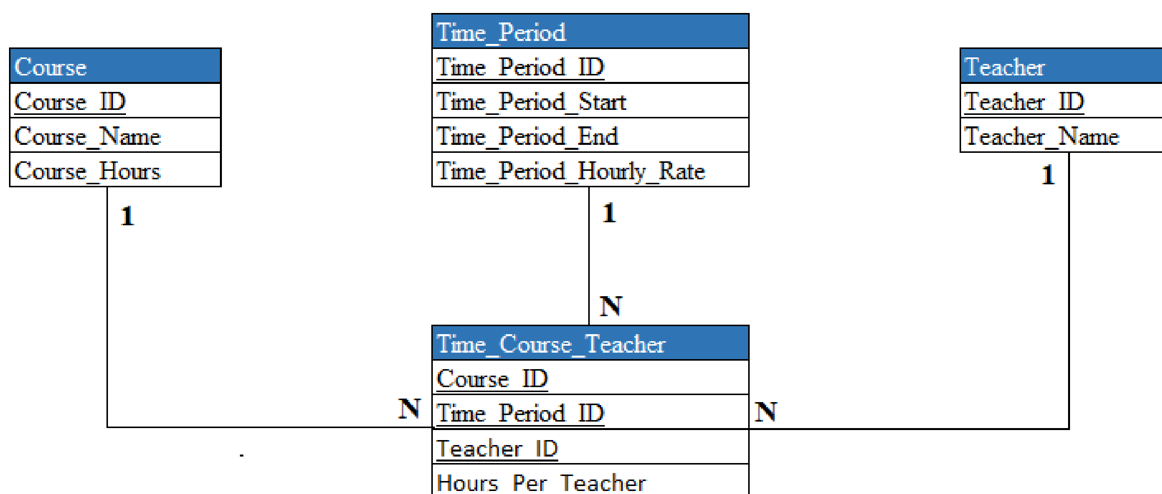
Част.1: Проектиране, моделиране и описание

1.1Създаване на ER (Entity-Relationship) модел



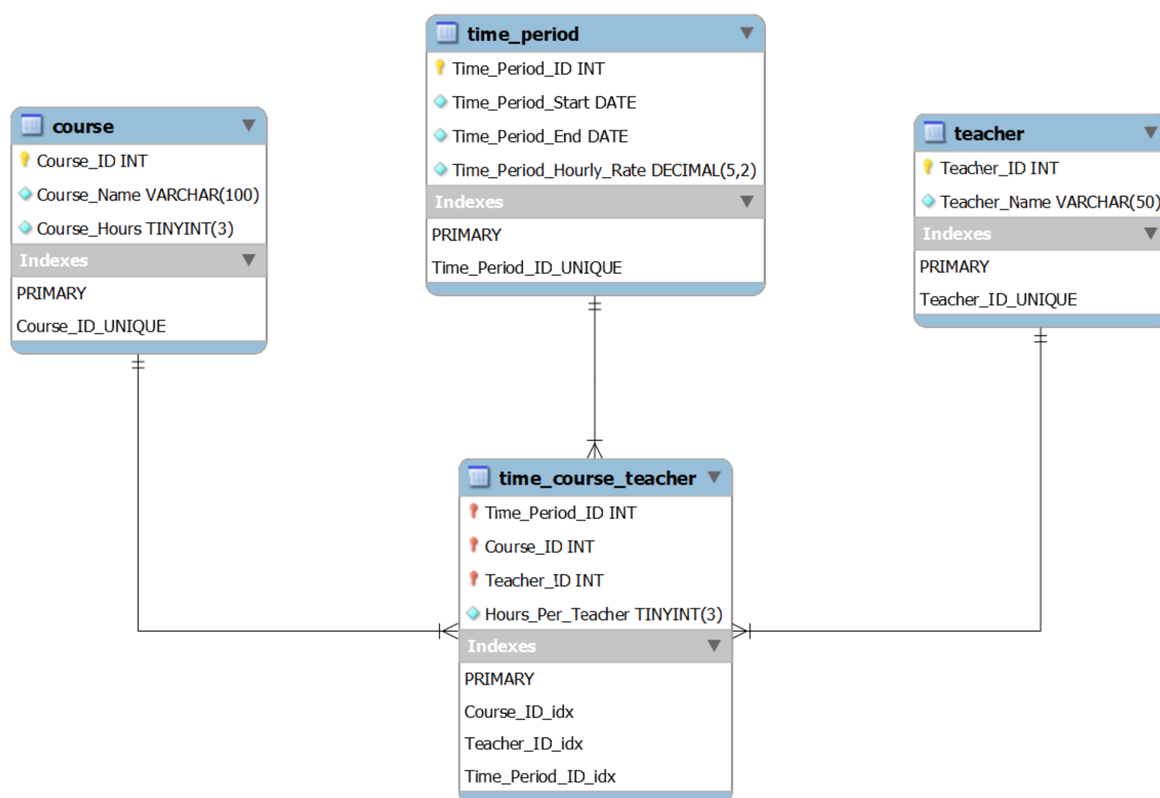
Фигура 1.1 представя ER модела в стил Чен на релационна база от данни

1.2Изготвяне на проект на релационната база от данни.



Фигура 1.2 Проект на релационна база данни.

1.3Разработване на EER (Enhanced Entity Relationship) модел



Фигура 1.3 показва разработеният EER модел базиран на ER модел на фигура 1.1.

1.4 Описание на таблиците и връзките между таблиците

1.4.1 Описание на таблиците

Table name	Column Name	Datatype	PK	NN	UQ	UN	AI	Default	FK	Referenced Table
Course	Course_ID	INT	✓	✓	✓		✓			
	Course_Name	VARCHAR(100)		✓						
	Course_Hours	TINYINT(3)		✓						

Таблица 1.1 показва цялата информация за table Course (това включва – различните колкони, техните типове данни и спецификациите зададени за всяка една колона).

Table name	Column Name	Datatype	PK	NN	UQ	UN	AI	Default	FK	Referenced Table
Time_Period	Time_Period_ID	INT	✓	✓	✓		✓			
	Time_Period_Start	DATE		✓						
	Time_Period_End	DATE		✓						
	Time_Period_Hourly_Rate	DECIMAL(5,2)		✓						

Таблица 1.2 Представя същата информация като таблица 1.1, но за table Time_Period.

Table name	Column Name	Datatype	PK	NN	UQ	UN	AI	Default	FK	Referenced Table
Teacher	Teacher_ID	INT	✓	✓	✓		✓			
	Teacher_Name	VARCHAR(50)		✓						

Таблица 1.3 Представя същата информация като таблица 1.1, но за table Teacher.

Table name	Column Name	Datatype	PK	NN	UQ	UN	AI	Default	FK	Referenced Table
Time_Course_Teacher	Course_ID	INT	✓	✓					✓	'курсов_проект'.'course'
	Time_Period_ID	INT	✓	✓					✓	'курсов_проект'.'time_period'
	Teacher_ID	INT	✓	✓					✓	'курсов_проект'.'teacher'
	Hours_Per_Teacher	TINYINT(3)		✓						

Таблица 1.4 Представя същата информация като таблица едно, но за table Time_Course_Teacher.

1.4.2 Описание на връзките между таблиците

Referencing Table	Cardinality	Referenced Table
Time_Course_Teacher Foreign Key: Course_ID Course_ID: INT (PK) Mandatory: Yes	One-to-Many (1:N) Identifying Relationship: Yes	Course Course_ID: INT(PK) Mandatory: Yes

Таблица 1.5 показва връзките между 2 различни таблици в базата от данни, в този случай е връзката между table Teacher и table Time_Coure_Teacher. Това ни дава различна информац като – външни ключове, приложение на вънчния ключ в наследяващата таблица, кардиналност на връзката, зависимост на таблиците една от друга и тн.

Referencing Table	Cardinality	Referenced Table
Time_Course_Teacher Foreign Key: Time_Period_ID Time_Period_ID: INT (PK) Mandatory: Yes	One-to-Many (1:N) Identifying Relationship: Yes	Time_Period Time_Period_ID: INT(PK) Mandatory: Yes

Таблица 1.6 ни дава същата информация като таблица 1.5, но този път между table Time_Period и table Time_Course_Teacher.

Referencing Table	Cardinality	Referenced Table
Time_Course_Teacher Foreign Key: Teacher_ID Teacher_ID: INT (PK) Mandatory: Yes	One-to-Many (1:N) Identifying Relationship: Yes	Teacher Teacher_ID: INT(PK) Mandatory: Yes

Таблица 1.7 отново предоставя същата информация като таблица 1.5 и 1.6, но този път между table Teacher и table Time_Course_Teacher.

Част.2: SQL – Създаване на базата от данни и таблиците към нея. Въвеждане на данни в таблиците

2.1 SQL заявка за създаване на базата от данни

```
CREATE DATABASE `курсов_проект` /*!40100 DEFAULT CHARACTER SET
utf8mb4 COLLATE utf8mb4_0900_ai_ci */ /*!80016 DEFAULT
ENCRYPTION='N' */;
```

Фигура 2.1 На фигура 2.1 е показан кодът за създаване на нашата База Данни в средата за разработка на БД – MySQL.

2.2 SQL заявки за създаване на всяка таблица от базата данни

```
CREATE TABLE `course` (

  `Course_ID` int NOT NULL AUTO_INCREMENT,

  `Course_Name` varchar(45) NOT NULL,

  `Course_Hours` tinyint NOT NULL,

  PRIMARY KEY (`Course_ID`),

  UNIQUE KEY `Course_ID_UNIQUE` (`Course_ID`)

) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```

Фигура 2.2 показва кодът за създаване на таблица ‘course’ и задаване на нейните колони и техните храктеристики.

```
CREATE TABLE `teacher` (

  `Teacher_ID` int NOT NULL AUTO_INCREMENT,

  `Teacher_Name` varchar(45) NOT NULL,

  PRIMARY KEY (`Teacher_ID`),

  UNIQUE KEY `Teacher_ID_UNIQUE` (`Teacher_ID`)

) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```

Фигура 2.3 показва кодът за създаване на таблица ‘teacher’ и задаване на нейните колони и техните храктеристики.

```
CREATE TABLE `time_period` (
```

```
  `Time_Period_ID` int NOT NULL AUTO_INCREMENT,
  `Time_Period_Start` date NOT NULL,
  `Time_Period_End` date NOT NULL,
  `Time_Period_Hourly_Rate` decimal(5,2) NOT NULL,
  PRIMARY KEY (`Time_Period_ID`),
  UNIQUE KEY `Time_Period_ID_UNIQUE` (`Time_Period_ID`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```

Фигура 2.4 показва кодът за създаване на таблица 'time_period' и задаване на нейните колони и техните храктеристики.

```
CREATE TABLE `time_course_teacher` (
  `Time_Period_ID` int NOT NULL,
  `Course_ID` int NOT NULL,
  `Teacher_ID` int NOT NULL,
  `Hours_Per_Teacher` tinyint NOT NULL,
  PRIMARY KEY (`Time_Period_ID`,`Course_ID`,`Teacher_ID`),
  KEY `Course_ID_idx` (`Course_ID`),
  KEY `Teacher_ID_idx` (`Teacher_ID`) /*!80000 INVISIBLE */,
  KEY `Time_Period_ID_idx` (`Time_Period_ID`),
  CONSTRAINT `Course_ID` FOREIGN KEY (`Course_ID`) REFERENCES
`course` (`Course_ID`),
  CONSTRAINT `Teacher_ID` FOREIGN KEY (`Teacher_ID`) REFERENCES
`teacher` (`Teacher_ID`),
  CONSTRAINT `Time_Period_ID` FOREIGN KEY (`Time_Period_ID`)
REFERENCES `time_period` (`Time_Period_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```

Фигура 2.5 показва кодът за създаване на таблица 'time_course_teacher' и задаване на нейните колони и техните храктеристики.

2.3 SQL заявки за въвеждане на данни във всяка една от таблиците

```
INSERT INTO Course (Course_Name, Course_Hours)
VALUES ('Amazon AWS', 42), ('Microsoft Azure', 36), ('Google
Cloud Platform', 30);
```

Фигура 2.6 представя въвеждане на данни в определено избрани колони със определено избрани стойности в вече създадената таблица ‘course’.

```
INSERT INTO Teacher (teacher_name)

VALUES ('Боряна Борисова'), ('Даниела Димитрова'), ('Елена Емилова');
```

Фигура 2.7 представя въвеждане на данни в определено избрани колони със определено избрани стойности в вече създадената таблица ‘teacher’.

```
INSERT INTO time_period (Time_Period_Start, Time_Period_End, Time_Period_Hourly_Rate)

VALUES ('2023-02-06', '2023-04-02', 120),

       ('2023-04-10', '2023-06-04', 140);
```

Фигура 2.8 представя въвеждане на данни в определено избрани колони със определено избрани стойности в вече създадената таблица ‘time_period’.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID, Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE Time_Period_Start = '2023-02-06' AND Time_Period_End = '2023-04-02'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Amazon AWS'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Боряна Борисова'),

  12);
```

Фигура 2.9 показва въвеждането на данни в определено избрани колони в таблицата ‘time_course_teacher’, но с подбрана информация извлечена чрез използване на SELECT от другите три таблици обвързани с тази.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID, Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE Time_Period_Start = '2023-02-06' AND Time_Period_End = '2023-04-02'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Microsoft Azure'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Боряна Борисова'),
```

```
  10);
```

Фигура 2.10 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID,
Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE
Time_Period_Start = '2023-04-10' AND Time_Period_End = '2023-06-
04'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Microsoft
Azure'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Боряна
Борисова'),

  11);
```

Фигура 2.11 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID,
Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE
Time_Period_Start = '2023-04-10' AND Time_Period_End = '2023-06-
04'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Google
Cloud Platform'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Боряна
Борисова'),

  16);
```

Фигура 2.12 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID,
Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE
Time_Period_Start = '2023-02-06' AND Time_Period_End = '2023-04-
02'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Amazon
AWS'),
```

```
  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Даниела
Димитрова'),

  30);
```

Фигура 2.13 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID,
Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE
Time_Period_Start = '2023-02-06' AND Time_Period_End = '2023-04-
02'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Google
Cloud Platform'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Даниела
Димитрова'),

  14);
```

Фигура 2.14 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID,
Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE
Time_Period_Start = '2023-04-10' AND Time_Period_End = '2023-06-
04'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Amazon
AWS'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Даниела
Димитрова'),

  13);
```

Фигура 2.15 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID,
Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE
Time_Period_Start = '2023-04-10' AND Time_Period_End = '2023-06-
04'),
```

```
  (SELECT Course_ID FROM Course WHERE course_name = 'Microsoft Azure'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Даниела Димитрова'),

  25);
```

Фигура 2.16 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID, Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE Time_Period_Start = '2023-02-06' AND Time_Period_End = '2023-04-02'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Microsoft Azure'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Елена Емилова'),

  26);
```

Фигура 2.17 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID, Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE Time_Period_Start = '2023-02-06' AND Time_Period_End = '2023-04-02'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Google Cloud Platform'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Елена Емилова'),

  16);
```

Фигура 2.18 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID, Teacher_ID, hours_per_teacher)

VALUES (
```

```
  (SELECT Time_period_ID FROM Time_period WHERE
Time_Period_Start = '2023-04-10' AND Time_Period_End = '2023-06-
04'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Amazon
AWS'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Елена
Емилова'),

  29);
```

Фигура 2.19 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

```
INSERT INTO Time_Course_Teacher (Time_period_ID, Course_ID,
Teacher_ID, hours_per_teacher)

VALUES (

  (SELECT Time_period_ID FROM Time_period WHERE
Time_Period_Start = '2023-04-10' AND Time_Period_End = '2023-06-
04'),

  (SELECT Course_ID FROM Course WHERE course_name = 'Google
Cloud Platform'),

  (SELECT Teacher_ID FROM Teacher WHERE teacher_name = 'Елена
Емилова'),

  14);
```

Фигура 2.20 визуализира същият подход като фиг.2.9, но с извличане и записване на различна информация чрез SELECT.

Част.3: SQL – Създаване на заявки по зададени условия и резултати от тяхното изпълнение

3.1 Пълни данни от всяка една таблица

```
SELECT * FROM Course ORDER BY Course_ID;
```

	Course_ID	Course_Name	Course_Hours
▶	1	Amazon AWS	42
	2	Microsoft Azure	36
	3	Google Cloud Platform	30
*	NULL	NULL	NULL

Course 27 ×

Output

Action Output

	#	Time	Action	Message
✔	1	02:35:42	SELECT * FROM Course ORDER BY Course_ID LIMIT 0, 1000	3 row(s) returned

Фигура 3.1 и Таблица 3.1 ни предоставят съответно кодът за извличане на цялата информация от таблица course и нейната подреба по Coure_Id (първичният ключ) и визуалицята на изпълненият код.

```
SELECT * FROM Time_period ORDER BY Time_period_ID;
```

Time_Period_ID	Time_Period_Start	Time_Period_End	Time_Period_Hourly_Rate
1	2023-02-06	2023-04-02	120.00
2	2023-04-10	2023-06-04	140.00
NULL	NULL	NULL	NULL

Time_period 28

Output

Action Output

#	Time	Action	Message
1	02:37:29	SELECT * FROM Time_period ORDER BY Time_period_ID LIMIT 0, ...	2 row(s) returned

Фигура 3.2 и Таблица 3.2 ни предоставят съответно кодът за извличане на цялата информация от таблица time_period и нейната подреба по Time_Period_ID (първичният ключ) и визуалицята на изпълненият код.

```
SELECT * FROM Teacher ORDER BY Teacher_ID;
```

Teacher_ID	Teacher_Name
1	Боряна Борисова
2	Даниела Димитрова
3	Елена Емилова
NULL	NULL

Teacher 30

Output

Action Output

#	Time	Action	Message
1	02:38:24	SELECT * FROM Teacher ORDER BY Teacher_ID LIMIT 0, 1000	3 row(s) returned

Фигура 3.3 и Таблица 3.3 ни предоставят съответно кодът за извличане на цялата информация от таблица teacher и нейната подреба по Teacher_ID (първичният ключ) и визуалицята на изпълненият код.

```
SELECT * FROM Time_course_teacher ORDER BY Time_Period_ID AND
Course_ID;
```

	Time_Period_ID	Course_ID	Teacher_ID	Hours_Per_Teacher
▶	1	1	1	12
	1	1	2	30
	1	2	1	10
	1	2	3	26
	1	3	2	14
	1	3	3	16
	2	1	2	13
	2	1	3	29
	2	2	1	11
	2	2	2	25
	2	3	1	16
	2	3	3	14
*	NULL	NULL	NULL	NULL

Time_course_teacher 31 ×

Output

Action Output

	#	Time	Action	Message
✔	1	02:39:02	SELECT * FROM Time_course_teacher ORDER BY Time_Period_ID AND Course_I...	12 row(s) returned

Фигура 3.4 и Таблица 3.4 ни предоставят съответно кодът за извличане на цялата информация от таблица Time_course_teacher и нейната подреба по първичните ключове Time_Period_ID и Course_ID и визуалицята на изпълнeният код.

3.2 Данни за курсовете за обучение

```
SELECT

    c.Course_ID,

    c.course_name,

    c.course_hours,

    CONCAT(FORMAT(SUM(c.course_hours *
tp.time_period_hourly_rate), 2, 'bg_BG'), ' лв.') AS
Total_Per_Teachers

FROM

    Course AS c

    INNER JOIN Time_period AS tp

GROUP BY
```

```
    c.Course_ID, c.course_name, c.course_hours;
```

	Course_ID	course_name	course_hours	Total_Per_Teachers
▶	1	Amazon AWS	42	10 920,00 лв.
	2	Microsoft Azure	36	9 360,00 лв.
	3	Google Cloud Platform	30	7 800,00 лв.

Result 52 ×

Output

Action Output

	#	Time	Action	Message
✔	1	03:14:31	SELECT c.Course_ID, c.course_name, c.course_hours, CONCAT(FORMAT(...	3 row(s) returned

Фигура 3.5 и Таблица 3.5 ни показват кодът за извличане на цялата информация от таблица course и създаване на допълнителна колона за пресмятане на получените пари от всички преподаватели в периодът на провеждане на курсовете. Фиг.3.10 е визуализацията на информацията, подредена по номер на курса.

```
SELECT c.Course_ID, c.Course_Name, c.Course_Hours,
tp.Time_Period_ID,

       DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') AS
Start_Date,

       DATE_FORMAT(tp.Time_Period_End, '%d.%m.%Y') AS End_Date,

       CONCAT(FORMAT(tp.Time_Period_Hourly_Rate, 2, 'bg_BG'), '
лв.') AS Time_Period_Hourly_Rate,

       CONCAT(FORMAT(ROUND(c.Course_Hours *
tp.Time_Period_Hourly_Rate), 2, 'bg_BG'), ' лв.') AS
total_per_teachers

FROM course AS c

INNER JOIN time_period AS tp

WHERE tp.Time_Period_ID IN (1, 2)

ORDER BY c.Course_ID, tp.Time_Period_ID;
```

	Course_ID	Course_Name	Course_Hours	Time_Period_ID	Start_Date	End_Date	Time_Period_Hourly_Rate	total_per_teachers
▶	1	Amazon AWS	42	1	06.02.2023	02.04.2023	120,00 лв.	5040.00 лв.
	1	Amazon AWS	42	2	10.04.2023	04.06.2023	140,00 лв.	5880.00 лв.
	2	Microsoft Azure	36	1	06.02.2023	02.04.2023	120,00 лв.	4320.00 лв.
	2	Microsoft Azure	36	2	10.04.2023	04.06.2023	140,00 лв.	5040.00 лв.
	3	Google Cloud Platform	30	1	06.02.2023	02.04.2023	120,00 лв.	3600.00 лв.
	3	Google Cloud Platform	30	2	10.04.2023	04.06.2023	140,00 лв.	4200.00 лв.

Result 34 ×

Output

Action Output

	#	Time	Action	Message
✔	1	02:44:22	SELECT c.Course_ID, c.Course_Name, c.Course_Hours, tp.Time_Period_ID, DATE...	6 row(s) returned

Фигура 3.6 и Таблица 3.6 ни предоставят съответно кодът за извличане на цялата информация от таблиците course и time_period и създаването на нова колона total_per_teacher, която този път ни дава информация за получените пари от преподавателите за различните курсове по време на тяхното провеждане в зависимост от това в кой период биват проведени. Резултатър бива подреден по Course_ID и Time_Period_ID.

```
SELECT

    c.Course_ID,

    c.Course_name,

    c.Course_hours,

    tp.Time_period_ID,

    DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') AS Start_date,

    DATE_FORMAT(tp.Time_Period_End, '%d.%m.%Y') AS End_date,

    CONCAT(FORMAT(tp.Time_Period_Hourly_Rate, 2, 'bg_BG'), '
лв.') AS Time_Period_Hourly_Rate,

    t.Teacher_ID, t.Teacher_Name,

    tct.Hours_per_teacher,

    CONCAT(FORMAT(tp.Time_period_hourly_rate *
tct.Hours_per_teacher, 2, 'bg_BG'), ' лв.') AS Total_per_teacher

FROM

    course AS c

    INNER JOIN time_course_teacher AS tct ON c.Course_ID =
tct.Course_ID

    INNER JOIN teacher AS t ON tct.Teacher_ID = t.Teacher_ID

    INNER JOIN time_period AS tp ON tct.Time_period_ID =
tp.Time_period_ID

ORDER BY

    c.Course_ID, tp.Time_period_ID, t.Teacher_ID;
```

Course_ID	Course_name	Course_hours	Time_period_ID	Start_date	End_date	Time_Period_Hourly_Rate	Teacher_ID	Teacher_Name	Hours_per_teacher	Total_per_teacher
1	Amazon AWS	42	1	06.02.2023	02.04.2023	120,00 лв.	1	Боряна Борисова	12	1440.00 лв.
1	Amazon AWS	42	1	06.02.2023	02.04.2023	120,00 лв.	2	Даниела Димитрова	30	3600.00 лв.
1	Amazon AWS	42	2	10.04.2023	04.06.2023	140,00 лв.	2	Даниела Димитрова	13	1820.00 лв.
1	Amazon AWS	42	2	10.04.2023	04.06.2023	140,00 лв.	3	Елена Емилова	29	4060.00 лв.
2	Microsoft Azure	36	1	06.02.2023	02.04.2023	120,00 лв.	1	Боряна Борисова	10	1200.00 лв.
2	Microsoft Azure	36	1	06.02.2023	02.04.2023	120,00 лв.	3	Елена Емилова	26	3120.00 лв.
2	Microsoft Azure	36	2	10.04.2023	04.06.2023	140,00 лв.	1	Боряна Борисова	11	1540.00 лв.
2	Microsoft Azure	36	2	10.04.2023	04.06.2023	140,00 лв.	2	Даниела Димитрова	25	3500.00 лв.
3	Google Cloud Platform	30	1	06.02.2023	02.04.2023	120,00 лв.	2	Даниела Димитрова	14	1680.00 лв.
3	Google Cloud Platform	30	1	06.02.2023	02.04.2023	120,00 лв.	3	Елена Емилова	16	1920.00 лв.
3	Google Cloud Platform	30	2	10.04.2023	04.06.2023	140,00 лв.	1	Боряна Борисова	16	2240.00 лв.
3	Google Cloud Platform	30	2	10.04.2023	04.06.2023	140,00 лв.	3	Елена Емилова	14	1960.00 лв.

Result 42 ×

Output

Action Output

#	Time	Action	Message
1	02:53:25	SELECT c.Course_ID, c.Course_name, c.Course_hours, tp.Time_period_ID,...	12 row(s) returned

Фигура 3.7 и Таблица 3.7 показват изваждането на цялата информация от всички възможни таблици е нейната визуалзация. Чрез използването на цялата тази информация можем да съставим нова колона total_per_teacher чрез която пресмятаме доходите на един преподавател въз основа на изработените му часове в записаните курсове в двата възможни периода на тяхното провеждане.

3.3 Данни за преподавателите, провеждащи курсовете за обучение

```
SELECT
    t.Teacher_ID,
    t.Teacher_name,
    SUM(tct.Hours_per_teacher) AS Total_hours,
    CONCAT(FORMAT(SUM(tct.Hours_per_teacher *
tp.Time_period_hourly_rate), 2, 'bg_BG'), ' лв.') AS
Total_income
FROM
    teacher AS t
    INNER JOIN time_course_teacher AS tct ON t.Teacher_ID =
tct.Teacher_ID
    INNER JOIN time_period AS tp ON tct.Time_period_ID =
tp.Time_period_ID
GROUP BY
    t.Teacher_ID;
```

	Teacher_ID	Teacher_name	Total_hours	Total_income
▶	1	Боряна Борисова	49	6 420,00 лв.
	2	Даниела Димитрова	82	10 600,00 лв.
	3	Елена Емилова	85	11 060,00 лв.

Result 44 ×

Output

Action Output

	#	Time	Action	Message
✔	1	02:56:46	SELECT t.Teacher_ID, t.Teacher_name, SUM(tct.Hours_per_teacher) AS Total...	3 row(s) returned

Фигура 3.8 и Таблица 3.8 ни дават информация за цялата таблица teacher плюс допълнително извеждане на две нови колони. Колоната Total_Hours ни дава представа за общият брой работни часове на съотетните преподаватели по време на всички периоди и всички курсове и колоната Total_income ни дава представа за всички доходи получени през тези часове.

```
SELECT
    t.Teacher_ID,
    t.Teacher_name,
    SUM(tct.Hours_per_teacher) AS Total_hours,
    tp.Time_period_ID,
    DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') AS Start_date,
    DATE_FORMAT(tp.Time_Period_End, '%d.%m.%Y') AS End_date,
    CONCAT(FORMAT(tp.Time_Period_Hourly_Rate, 2, 'bg_BG'), '
лв.') AS Hourly_Rate,
    CONCAT(FORMAT(SUM(tct.Hours_per_teacher *
tp.Time_period_hourly_rate), 2, 'bg_BG'), ' лв.') AS
Total_income
FROM
    teacher AS t
    INNER JOIN time_course_teacher AS tct ON t.Teacher_ID =
tct.Teacher_ID
    INNER JOIN time_period AS tp ON tct.Time_period_ID =
tp.Time_period_ID
WHERE
    tct.Time_period_ID IN (1, 2)
```

```
GROUP BY
    t.Teacher_ID,
    tp.Time_period_ID
ORDER BY
     t.Teacher_ID,
    tp.Time_Period_ID;
```

	Teacher_ID	Teacher_name	Total_hours	Time_period_ID	Start_date	End_date	Hourly_Rate	Total_income
▶	1	Боряна Борисова	22	1	06.02.2023	02.04.2023	120,00 лв.	2 640,00 лв.
	1	Боряна Борисова	27	2	10.04.2023	04.06.2023	140,00 лв.	3 780,00 лв.
	2	Даниела Димитрова	44	1	06.02.2023	02.04.2023	120,00 лв.	5 280,00 лв.
	2	Даниела Димитрова	38	2	10.04.2023	04.06.2023	140,00 лв.	5 320,00 лв.
	3	Елена Емилова	42	1	06.02.2023	02.04.2023	120,00 лв.	5 040,00 лв.
	3	Елена Емилова	43	2	10.04.2023	04.06.2023	140,00 лв.	6 020,00 лв.

Result 46 ×

Output

Action Output

#	Time	Action	Message
1	02:59:18	SELECT t.Teacher_ID, t.Teacher_name, SUM(tct.Hours_per_teacher) AS Total...	6 row(s) returned

Фигура 3.9 и Таблица 3.9 илюстрират използването на информацията от таблиците teacher и time_period, с помощта на две нови колони total_hours и total_income. Чрез колоната total_hours можем да видим изработените часове на преподавателят в даденият период за провеждане на курсове, докато чрез колоната total_income можем да проверим доходите на същият преподавател чрез изработените часове в определеният период.

```
SELECT
    t.Teacher_ID,
    t.Teacher_name,
    tct.Hours_per_teacher,
    tp.Time_Period_ID,
    DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') AS Start_Date,
    DATE_FORMAT(tp.Time_Period_End, '%d.%m.%Y') AS End_Date,
    CONCAT(FORMAT(tp.Time_Period_Hourly_Rate, 2, 'bg_BG'), '
лв.') AS Hourly_Rate,
    c.Course_ID,
    c.Course_Name,
    CONCAT(FORMAT(tct.Hours_per_teacher *
tp.Time_Period_Hourly_Rate, 2, 'bg_BG'), ' лв.') AS Total_income
```

```sql
FROM
    teacher AS t
    INNER JOIN time_course_teacher AS tct ON t.Teacher_ID =
tct.Teacher_ID
    INNER JOIN time_period AS tp ON tct.Time_Period_ID =
tp.Time_Period_ID
    INNER JOIN course AS c ON tct.Course_ID = c.Course_ID
ORDER BY
    t.Teacher_ID,
    tp.Time_Period_ID,
    c.Course_ID;
```

Teacher_ID	Teacher_name	Hours_per_teacher	Time_Period_ID	Start_Date	End_Date	Hourly_Rate	Course_ID	Course_Name	Total_income
1	Боряна Борисова	12	1	06.02.2023	02.04.2023	120,00 лв.	1	Amazon AWS	1 440,00 лв.
1	Боряна Борисова	10	1	06.02.2023	02.04.2023	120,00 лв.	2	Microsoft Azure	1 200,00 лв.
1	Боряна Борисова	11	2	10.04.2023	04.06.2023	140,00 лв.	2	Microsoft Azure	1 540,00 лв.
1	Боряна Борисова	16	2	10.04.2023	04.06.2023	140,00 лв.	3	Google Cloud Platform	2 240,00 лв.
2	Даниела Димитрова	30	1	06.02.2023	02.04.2023	120,00 лв.	1	Amazon AWS	3 600,00 лв.
2	Даниела Димитрова	14	1	06.02.2023	02.04.2023	120,00 лв.	3	Google Cloud Platform	1 680,00 лв.
2	Даниела Димитрова	13	2	10.04.2023	04.06.2023	140,00 лв.	1	Amazon AWS	1 820,00 лв.
2	Даниела Димитрова	25	2	10.04.2023	04.06.2023	140,00 лв.	2	Microsoft Azure	3 500,00 лв.
3	Елена Емилова	26	1	06.02.2023	02.04.2023	120,00 лв.	2	Microsoft Azure	3 120,00 лв.
3	Елена Емилова	16	1	06.02.2023	02.04.2023	120,00 лв.	3	Google Cloud Platform	1 920,00 лв.
3	Елена Емилова	29	2	10.04.2023	04.06.2023	140,00 лв.	1	Amazon AWS	4 060,00 лв.
3	Елена Емилова	14	2	10.04.2023	04.06.2023	140,00 лв.	3	Google Cloud Platform	1 960,00 лв.

Result 48 ×

Output

Action Output

#	Time	Action	Message
1	03:02:02	SELECT t.Teacher_ID, t.Teacher_name, tct.Hours_per_teacher, tp.Time_Peri...	12 row(s) returned

Фигура 3.10 и Таблица 3.10 чрез използване на информацията от всички възможни таблици ни дават възможноста да пресметнем заплатата на всеки един преподавател за всичките изработени часове в техните курсове за всеки един от периодите за тяхното провеждане.

3.4 Данни за периодите от време, през които се провеждат курсовете за обучение

```sql
SELECT
    tp.Time_Period_ID,
    DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') AS Start_Date,
    DATE_FORMAT(tp.Time_Period_End, '%d.%m.%Y') AS End_Date,
```

```
    tp.Time_Period_Hourly_Rate,
    c.Course_ID,
    c.Course_name,
    c.Course_hours,
    CONCAT(FORMAT(c.Course_hours * tp.Time_Period_Hourly_Rate,
2, 'bg_BG'), ' лв.') AS total_per_teacher
FROM
    time_period AS tp
    INNER JOIN course AS c
ORDER BY
     tp.Time_Period_ID,
     c.Course_ID;
```

Time_Period_ID	Start_Date	End_Date	Time_Period_Hourly_Rate	Course_ID	Course_name	Course_hours	total_per_teacher
1	06.02.2023	02.04.2023	120.00	1	Amazon AWS	42	5 040,00 лв.
1	06.02.2023	02.04.2023	120.00	2	Microsoft Azure	36	4 320,00 лв.
1	06.02.2023	02.04.2023	120.00	3	Google Cloud Platform	30	3 600,00 лв.
2	10.04.2023	04.06.2023	140.00	1	Amazon AWS	42	5 880,00 лв.
2	10.04.2023	04.06.2023	140.00	2	Microsoft Azure	36	5 040,00 лв.
2	10.04.2023	04.06.2023	140.00	3	Google Cloud Platform	30	4 200,00 лв.

Result 54 ×

Output

Action Output

#	Time	Action	Message
1	03:23:31	SELECT tp.Time_Period_ID, DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') ...	6 row(s) returned

Фигура 3.11 и Таблица 3.11 ни предоставят чрез използване на информацията от таблиците time_period и course възможните доходи за преподавателите в зависимост от продължението на курсовете в различнит епериоди на тяхното провеждане.

```
SELECT
    tp.Time_Period_ID,
    DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') AS Start_Date,
    DATE_FORMAT(tp.Time_Period_End, '%d.%m.%Y') AS End_Date,
    tp.Time_Period_Hourly_Rate,
    t.Teacher_ID,
    t.Teacher_name,
    SUM(tct.Hours_per_teacher) AS Total_hours,
```

```
    CONCAT(FORMAT(SUM(tct.Hours_per_teacher *
tp.Time_Period_Hourly_Rate), 2, 'bg_BG'), ' лв.') AS
Total_income

FROM

    time_period AS tp

    INNER JOIN time_course_teacher AS tct ON tp.Time_Period_ID =
tct.Time_Period_ID

    INNER JOIN teacher AS t ON tct.Teacher_ID = t.Teacher_ID

GROUP BY

    tp.Time_Period_ID,

    Start_Date,

    End_Date,

    tp.Time_Period_Hourly_Rate,

    t.Teacher_ID,

    Teacher_name

ORDER BY

     tp.Time_Period_ID,

     t.Teacher_ID;
```

Time_Period_ID	Start_Date	End_Date	Time_Period_Hourly_Rate	Teacher_ID	Teacher_name	Total_hours	Total_income
1	06.02.2023	02.04.2023	120.00	1	Боряна Борисова	22	2 640,00 лв.
1	06.02.2023	02.04.2023	120.00	2	Даниела Димитрова	44	5 280,00 лв.
1	06.02.2023	02.04.2023	120.00	3	Елена Емилова	42	5 040,00 лв.
2	10.04.2023	04.06.2023	140.00	1	Боряна Борисова	27	3 780,00 лв.
2	10.04.2023	04.06.2023	140.00	2	Даниела Димитрова	38	5 320,00 лв.
2	10.04.2023	04.06.2023	140.00	3	Елена Емилова	43	6 020,00 лв.

Result 58 ×

Output


Action Output

#	Time	Action	Message
1	03:25:48	SELECT tp.Time_Period_ID, DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') ...	6 row(s) returned

Фигура 3.12 и Таблица 3.12 показват чрез използването на таблиците time_period и teacher и създаването на две нови колони покрай тях total_income и total_hours получените пари за различни преподаватели в зависимост от изработените часове без значение в кой курс в двата възможни периода на тяхното провеждане.

```
SELECT

    tp.Time_Period_ID,

    DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') AS Start_Date,
```

```
    DATE_FORMAT(tp.Time_Period_End, '%d.%m.%Y') AS End_Date,

    CONCAT(FORMAT(tp.Time_Period_Hourly_Rate, 2, 'bg_BG'), '
лв.') AS Hourly_Rate,

    c.Course_ID,

    c.Course_name,

    c.Course_hours,

    t.Teacher_ID,

    t.Teacher_name,

    tct.Hours_per_teacher,

    CONCAT(FORMAT(SUM(tct.Hours_per_teacher *
tp.Time_Period_Hourly_Rate), 2, 'bg_BG'), 'лв') AS
total_per_teacher

FROM

    time_period AS tp

    INNER JOIN time_course_teacher AS tct ON tp.Time_Period_ID =
tct.Time_Period_ID

    INNER JOIN course AS c ON tct.Course_ID = c.Course_ID

    INNER JOIN teacher AS t ON tct.Teacher_ID = t.Teacher_ID

GROUP BY

    tp.Time_Period_ID,

    Start_Date,

    End_Date,

    tp.Time_Period_Hourly_Rate,

    c.Course_ID,

    c.Course_name,

    c.Course_hours,

    t.Teacher_ID,

    Teacher_name

ORDER BY

     tp.Time_Period_ID,

    c.Course_id,
```

```
    t.Teacher_ID;
```

Time_Period_ID	Start_Date	End_Date	Hourly_Rate	Course_ID	Course_name	Course_hours	Teacher_ID	Teacher_name	Hours_per_teacher	total_per_teacher
1	06.02.2023	02.04.2023	120,00 лв.	1	Amazon AWS	42	1	Боряна Борисова	12	1 440,00лв
1	06.02.2023	02.04.2023	120,00 лв.	1	Amazon AWS	42	2	Даниела Димитрова	30	3 600,00лв
1	06.02.2023	02.04.2023	120,00 лв.	2	Microsoft Azure	36	1	Боряна Борисова	10	1 200,00лв
1	06.02.2023	02.04.2023	120,00 лв.	2	Microsoft Azure	36	3	Елена Емилова	26	3 120,00лв
1	06.02.2023	02.04.2023	120,00 лв.	3	Google Cloud Platform	30	2	Даниела Димитрова	14	1 680,00лв
1	06.02.2023	02.04.2023	120,00 лв.	3	Google Cloud Platform	30	3	Елена Емилова	16	1 920,00лв
2	10.04.2023	04.06.2023	140,00 лв.	1	Amazon AWS	42	2	Даниела Димитрова	13	1 820,00лв
2	10.04.2023	04.06.2023	140,00 лв.	1	Amazon AWS	42	3	Елена Емилова	29	4 060,00лв
2	10.04.2023	04.06.2023	140,00 лв.	2	Microsoft Azure	36	1	Боряна Борисова	11	1 540,00лв
2	10.04.2023	04.06.2023	140,00 лв.	2	Microsoft Azure	36	2	Даниела Димитрова	25	3 500,00лв
2	10.04.2023	04.06.2023	140,00 лв.	3	Google Cloud Platform	30	1	Боряна Борисова	16	2 240,00лв
2	10.04.2023	04.06.2023	140,00 лв.	3	Google Cloud Platform	30	3	Елена Емилова	14	1 960,00лв

Result 60

Output

Action Output

#	Time	Action	Message
1	03:28:42	SELECT tp.Time_Period_ID, DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') ...	12 row(s) returned

Фигура 3.13 и Таблица 3.13 ни дават информация чрез използването на цялата достъпна информация от всички възможни таблици за парите получени от отделените преподаватели за един и същ курс в зависимост от техните работни часове и периодът в който той е бил провед.

```
SELECT
    tp.Time_Period_ID,
    DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') AS Start_Date,
    DATE_FORMAT(tp.Time_Period_End, '%d.%m.%Y') AS End_Date,
    tp.Time_Period_Hourly_Rate,
    MIN(tct.Hours_per_teacher) AS MIN_Work_Hours,
    MAX(tct.Hours_per_teacher) AS MAX_Work_Hours,
    CONCAT(FORMAT(MIN(tct.Hours_per_teacher) *
tp.Time_Period_Hourly_Rate, 2, 'bg_BG'), ' лв.') AS
MIN_Earned_Amount,
    CONCAT(FORMAT(MAX(tct.Hours_per_teacher) *
tp.Time_Period_Hourly_Rate, 2, 'bg_BG'), ' лв.') AS
MAX_Earned_Amount
FROM
    time_period AS tp
    INNER JOIN time_course_teacher AS tct ON tp.Time_Period_ID =
tct.Time_Period_ID
GROUP BY
    tp.Time_Period_ID,
```

```
    Start_Date,
    End_Date,
    tp.Time_Period_Hourly_Rate;
```

Time_Period_ID	Start_Date	End_Date	Time_Period_Hourly_Rate	Min_Work_Hours	Max_Work_Hours	Min_Earned_Amount	Max_Earned_Amount
1	06.02.2023	02.04.2023	120.00	10	30	1 200,00 лв.	3 600,00 лв.
2	10.04.2023	04.06.2023	140.00	11	29	1 540,00 лв.	4 060,00 лв.

Result 62

Output

Action Output

#	Time	Action	Message
1	03:31:28	SELECT tp.Time_Period_ID, DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') ...	2 row(s) returned

Фигура 3.14 и Таблица 3.14 чрез използване напълно само от таблицата time_period и няколко допълнително изградени колони ни дават възможността да пресметмен най-малият и най-голямият доход измежду всички преподаватели въз основа на най-малко или най-много изработените часове на по време на двата отелни периода.

3.5 Съхранени процедури

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`courseDataAndStatus`(IN input_date DATE)
BEGIN
    SELECT
        c.Course_ID,
        c.Course_name,
        tp.Time_Period_ID,
        DATE_FORMAT(tp.Time_Period_Start, '%d.%m.%Y') AS
Time_Period_Start,
        DATE_FORMAT(tp.Time_Period_End, '%d.%m.%Y') AS
Time_Period_End,
        CASE
            WHEN input_date < tp.Time_Period_Start THEN
'Предстоящ'
            WHEN input_date >= tp.Time_Period_Start AND
input_date <= tp.Time_Period_End THEN 'Текущ'
```

```
            WHEN input_date > tp.Time_Period_End THEN
'Приключил'

        END AS Status

    FROM

        course AS c

        INNER JOIN time_course_teacher AS tct ON c.Course_ID =
tct.Course_ID

        INNER JOIN time_period AS tp ON tct.Time_Period_ID =
tp.Time_Period_ID

    GROUP BY

        c.Course_ID,

        tp.Time_Period_ID

     ORDER BY

          tp.Time_Period_ID,

        c.Course_ID;

END$$

DELIMITER ;
```

Фигура 3.15 изобразява код на съхранена процедура. Наща процедура работи единствено когато на нея бъде подадена определена входна стойност под формата на дата. Когато подадем тази входна стойност на процедурата тя започва да прави сравнения с датите които има съхранени като информация в таблците в базата данни. След проверката тя ни връща резултат който ни дава информация за това дали един курс е предстоящ, текущ или приключил.

```
CALL `courseDataAndStatus`('2023-01-22');
```

Course_ID	Course_name	Time_Period_ID	Time_Period_Start	Time_Period_End	Status
1	Amazon AWS	1	06.02.2023	02.04.2023	Предстоящ
2	Microsoft Azure	1	06.02.2023	02.04.2023	Предстоящ
3	Google Cloud Platform	1	06.02.2023	02.04.2023	Предстоящ
1	Amazon AWS	2	10.04.2023	04.06.2023	Предстоящ
2	Microsoft Azure	2	10.04.2023	04.06.2023	Предстоящ
3	Google Cloud Platform	2	10.04.2023	04.06.2023	Предстоящ

Result 64 ×

Output

Action Output

#	Time	Action	Message
1	03:47:52	CALL `courseDataAndStatus`('2023-01-22')	6 row(s) returned

Фигура 3.16 и Таблица 3.15 ни показват информацията за въведената дата след като тя е била ставнена с датите за двата периода за провеждане на курсовете.

```
CALL `courseDataAndStatus`('2023-02-22');
```

Course_ID	Course_name	Time_Period_ID	Time_Period_Start	Time_Period_End	Status
1	Amazon AWS	1	06.02.2023	02.04.2023	Текущ
2	Microsoft Azure	1	06.02.2023	02.04.2023	Текущ
3	Google Cloud Platform	1	06.02.2023	02.04.2023	Текущ
1	Amazon AWS	2	10.04.2023	04.06.2023	Предстоящ
2	Microsoft Azure	2	10.04.2023	04.06.2023	Предстоящ
3	Google Cloud Platform	2	10.04.2023	04.06.2023	Предстоящ

Result 65

Output

Action Output

#	Time	Action	Message
1	03:50:53	CALL `courseDataAndStatus`('2023-02-22')	6 row(s) returned

Фигура 3.17 и Таблица 3.16 ни показват информацията за въведената дата след като тя е била ставнена с датите за двата периода за провеждане на курсовете.

```
CALL `courseDataAndStatus`('2023-03-22');
```

Course_ID	Course_name	Time_Period_ID	Time_Period_Start	Time_Period_End	Status
1	Amazon AWS	1	06.02.2023	02.04.2023	Текущ
2	Microsoft Azure	1	06.02.2023	02.04.2023	Текущ
3	Google Cloud Platform	1	06.02.2023	02.04.2023	Текущ
1	Amazon AWS	2	10.04.2023	04.06.2023	Предстоящ
2	Microsoft Azure	2	10.04.2023	04.06.2023	Предстоящ
3	Google Cloud Platform	2	10.04.2023	04.06.2023	Предстоящ

Result 73

Output

Action Output

#	Time	Action	Message
1	03:52:57	CALL `courseDataAndStatus`('2023-03-22')	6 row(s) returned

Фигура 3.18 и Таблица 3.17 ни показват информацията за въведената дата след като тя е била ставнена с датите за двата периода за провеждане на курсовете.

```
CALL `courseDataAndStatus`('2023-04-22');
```

	Course_ID	Course_name	Time_Period_ID	Time_Period_Start	Time_Period_End	Status
▶	1	Amazon AWS	1	06.02.2023	02.04.2023	Приключил
	2	Microsoft Azure	1	06.02.2023	02.04.2023	Приключил
	3	Google Cloud Platform	1	06.02.2023	02.04.2023	Приключил
	1	Amazon AWS	2	10.04.2023	04.06.2023	Текущ
	2	Microsoft Azure	2	10.04.2023	04.06.2023	Текущ
	3	Google Cloud Platform	2	10.04.2023	04.06.2023	Текущ

Result 75 ×

Output

Action Output

	#	Time	Action	Message
✔	1	03:53:25	CALL 'courseDataAndStatus'('2023-04-22')	6 row(s) returned

Фигура 3.19 и Таблица 3.18 ни показват информацията за въведената дата след като тя е била ставнена с датите за двата периода за провеждане на курсовете.

```
CALL `courseDataAndStatus`('2023-05-22');
```

	Course_ID	Course_name	Time_Period_ID	Time_Period_Start	Time_Period_End	Status
▶	1	Amazon AWS	1	06.02.2023	02.04.2023	Приключил
	2	Microsoft Azure	1	06.02.2023	02.04.2023	Приключил
	3	Google Cloud Platform	1	06.02.2023	02.04.2023	Приключил
	1	Amazon AWS	2	10.04.2023	04.06.2023	Текущ
	2	Microsoft Azure	2	10.04.2023	04.06.2023	Текущ
	3	Google Cloud Platform	2	10.04.2023	04.06.2023	Текущ

Result 76 ×

Output

Action Output

	#	Time	Action	Message
✔	1	03:53:45	CALL 'courseDataAndStatus'('2023-05-22')	6 row(s) returned

Фигура 3.20 и Таблица 3.19 ни показват информацията за въведената дата след като тя е била ставнена с датите за двата периода за провеждане на курсовете.

```
CALL `courseDataAndStatus`(NULL);
```

Course_ID	Course_name	Time_Period_ID	Time_Period_Start	Time_Period_End	Status
1	Amazon AWS	1	06.02.2023	02.04.2023	NULL
2	Microsoft Azure	1	06.02.2023	02.04.2023	NULL
3	Google Cloud Platform	1	06.02.2023	02.04.2023	NULL
1	Amazon AWS	2	10.04.2023	04.06.2023	NULL
2	Microsoft Azure	2	10.04.2023	04.06.2023	NULL
3	Google Cloud Platform	2	10.04.2023	04.06.2023	NULL

Result 78

Output

Action Output

#	Time	Action	Message
1	03:54:14	CALL `courseDataAndStatus`(NULL)	6 row(s) returned

Фигура 3.16 и Таблица 3.15 ни показват информацията за въведена стойност различна от вече очакваната дата (в този случай е стойноста NULL).